Bioinformatics
doi.10.1093/bioinformatics/comgi10courseworkucabpev
Advance Access Publication Date: 03 March 2018
Manuscript Category: Coursework

OXFORD

Protein location

Predicting subcellular location of eukaryotic proteins using an ensemble of learners.

Pierre Eugene Valassakis [1,*]

[1] UCL Computer Science, London, WC1E 7JE, UK

[*] To whom correspondence should be addressed.

Associate Editor: Pierre Eugene Valassakis

Received on 21/03/2018; revised on 21/03/2018; accepted on 21/03/2018

Abstract

Motivation: With the stellar success of high-throughput gene sequencing and the complete reconstruction of thousands of genomes in the recent years, the question of gene function is evermore prevalent [5, 2]. Understanding the subcellular location of proteins is very interesting in this context, as it can lead to significant insights on the function of these proteins and by extension on the function their encoding genes [14]. Here we propose a method for computationally deriving protein location using an ensemble of known classifiers, and thanks to protein sequencing data and known properties of amino-acids.
Results: We have used the COMPGI10 Protein dataset [2] to train a classifier to discriminate between (1) Cytosolic, (2) Mitochondrial, (3) Nucleic and (4) Secreted proteins. Our ensemble classifier was composed of a Random Forest, a Gradient Boosting, a SVM and a Logistic Regression model. The set of features we used were based on well-known peptide properties such as hydrophobicity, isoelectric-point, side-chain charge, molecular weight and secondary-structure. We obtained an overall accuracy of 68% on the left out test set and a cross-validation score of $66\% \pm 3\%$.
Availability: Always.
Contact: ucabpev@ucl.ac.uk
Supplementary information: Available upon request.

1 Introduction

1.1 Task Overview and Related Studies

Determining protein function is of wide interest in a diverse set of fields, with applications ranging from proteomics and understanding signalling pathways in a cell to medicine and the study of various diseases [12, 14]. This is accentuated with the recent advances in high-throughput genomics and the constitution of thousands of complete genomes, as determining the function of different proteins can lead to significant insights to the functions of the corresponding genes [14, 5]. Again, many fields benefit from such knowledge, from the study and understanding of genetic diseases to evolutionary biology and the study of the processes leading to the diversity of species observed on earth today.

In determining protein function, a good indicator is protein location within the cell [6]. While a Prokaryotic cell structure is fairly simple, Eukaryotic cells are highly compartmentalised with more than 10 identified main subcellular locations for proteins [14]. In predicting protein location, different approaches have been proposed over the years [14], and in this paper we focus on using aminoacid sequence composition as our main predictor. In this context, early work was conducted by [19], and only discriminated between intracellular and secreted proteins [14]. A plethora of new studies have emerged since, using a variety of different algorithms for their predictions. Main differentiating points on these studies are (1) performance of the models, and (2) granularity in terms of how many protein locations are considered [14, 6, 12, 22]. Current state-of-the-art results arrive at accuracies above 70%, considering more than 10 possible location sites [22].

1.2 Background

1.2.1 Protein locations and aminoacid properties considered.

In our work, we investigate discriminating between four classes of Eukaryotic proteins, namely (1) Cytosolic, (2) Mitochondrial, (3) Nucleic and (4) Secreted. Cytosolic proteins reside in the cytosol, the main fluid constituting the cytoplasm of a cell [21]. Mitochondrial proteins reside in the Mitochondria, which are organelles responsible for converting consumed nutrients into energy necessary for the functioning of the cell [18]. Nucleic proteins reside in the Nucleus, the part of the cell that carries its genetic code [9]. Secreted proteins are proteins that leave the cell by traversing its plasma membrane [20]. In building our models, we consider a number of aminoacid properties which we use in various setups. These properties are enumerated in Table 1, along with a brief explanation of what they represent.

Table 1. Description of aminoacid and polypeptide properties used in our models.

Property	**Description**
Isoelectic point	The isoelectric point of an aminoacid is the Ph value at which it is neutrally charged [16].
Hydrophobicity	The hydrophobicity of an aminoacid is a measure of the amount it is repelled by water. A negative hydrophobicity indicates that the molecule is hydrophilic, i.e attracted by water [17, 7].
Molecular Weight	The molecular weight of an aminoacid is the mass of one mole of that aminoacid [11].
Secondary Structure	The secondary structure of a polypeptide chain refers to the spacial arrangement of its aminoacids, which are generally found in α-helix, β-sheet and turn arrangements [8].
Aromaticity	The aromaticity of a polypeptide chain is the relative frequency of aromatic aminoacids within it. Aromatic aminoacids are those that contain an aromatic ring within their structure [7].
Side chain charge	The side chain charge of an aminoacid refers to the electric charge carried by its side chain [1].
Instability index	The instability index measures the stability of the polypeptide chain, i.e. how prone it is to denaturing [13].

1.2.2 Relevant machine learning classifiers

In building our solution, we used four popular machine learning classifiers, namely Random Forests, Gradient Boosting, Logistic Regression and Support Vector Machines (SVMs). It is interesting to briefly review the core ideas behind these techniques.

Random Forest and Gradient Boosting classifiers both consist in ensembling decision trees to make their predictions. Decision trees are a type of classifier that sequentially split the data according to their most discriminative features. In other words, during training, a decision tree will find what is the best predictor and the corresponding threshold value for splitting the data between the target classes followed by the second best predictor and so on. During inference, classification happens by moving down the tree, comparing the value of each feature of the example considered with the thresholds found on each node, and the final predicted class is the class of the end leaf. The difference between Gradient Boosting and Random Forests then is in the way they ensemble the decision trees they use, with Random Forests using a technique known as bagging while Gradient boosting uses the boosting method [15].

SVMs are a type of classifier that aim to find a decision boundary in feature space that maximizes the margin between data-points. In order to do so they often rely on kernel functions to better separate non-linearly separable data. During inference, SVMs look at which side of the decision boundary the new data point lies to make the classification decision. Although based on a different principle, logistic regression classifiers are quite similar to SVMs with a linear kernel function. Specifically, logistic regression models are trained to maximize the likelihood of the data based on probabilities obtained by the logistic function (rather than maximising the separation margin between the data). In inference however, these classifiers also apply a decision boundary (albeit a linear one) on the data [15].

2 Approach

As we stated above, we are using the COMPGI10 dataset [2] to build a classifier capable of discriminating between proteins that lie in the Cytosol, the Mictochondria, the Nucleus, and that are Secreted.

Our approach consisted in several steps, enumerated below and detailed in the following sections:

1. Preprocessing the data and building the features used for classification: We split the available data to a training and a test set, and converted the raw string sequence data into a meaningful vector in feature space, by using known aminoacid properties.
2. Building the model: Our model consists of an ensemble of a SVM, a random forest, a gradient boosting and a logistic regression classifier that arrive to consensus on classification by a soft voting mechanism.
3. Quantifying performance and building error analytics: With our models trained and tuned, we obtained a series of metrics for performance. We calculated the accuracy, precision, recall and F1 score both using 20-fold cross-validation on the training data and by using the left out test set. We also plot Receiver Operating Characteristic (ROC) curves and a confusion matrix, and attempt to find meaningful patterns on the misclassified data.

In the following sections, we go through our methods in detail, and engage in a discussion analysing our results.

3 Methods

The language we used for our experiments was python [3]. All our models, cross-validation setups and error analysis are made using Scikit-learn, which is a python machine learning library [4]. To process the biological data and extract meaningful information from there we also used the Biopython library [10] .

3.1 Preprocessing and Feature selection

3.1.1 Preprocessing

The COMPGI10 dataset is made of 9222 protein sequences in FASTA format, divided into four folders that represent each of our classes. The first step in our approach consisted in transforming the data into a usable format : We loaded and concatenated the string sequences into one long list, and created a corresponding list of labels. For the labels, our numerical encoding and accepted shorthand notation are depicted in Table 2.

With our data in this format, we proceeded to split them into a training set and a test set. The test set is made of 10% of randomly chosen data, and is left out of any training or tuning process, only to be used in the last performance assessment of our model.

Table 2. Numerical encoding and abbreviations for the labels

Label String	Abbreviation	Label Encoding
Cytosolic	Cyto	0
Mitochondrial	Mito	1
Nucleic	Nuc	2
Secreted	Secr	3

3.1.2 Feature selection

With traditional machine learning models, sequence data is typically not well suited for discriminating between polypeptides, and hand-crafted features need to be created. Our feature processing module then takes as an input the sequence string and outputs a numerical vector representing the sequence in feature space. In order to do so, we rely on Biopython's ProteinAnalysis Class [10], as well as Kyte and Doolittle's hydrophobicity table [17] and side chain charge tables [23].

Overall, the features we used are the following:

- Polypeptide sequence length.
- Aminoacid composition (how many of each aminoacid there are, as a fraction of the total number of aminoacids).
- Bipeptide composition (how many of each aminoacid pairs there are).
- Total hydrophobic score (sum of positive hydrophobicities).
- Total hydrophilic score (sum of negative hydrophobicities).
- Average hydrophobicity.
- Total positive charge (sum of positive side chain charges).
- Total negative charge (sum of negative side chain charges).
- Average side chain charge.
- Molecular weight of the polypeptide.
- Secondary structure features (three features representing the fraction of aminoacids that tend to be part of helices, sheets or turns).
- Aromaticity of the chain.
- Instability index of the chain.

Extra features were also created that consisted of the above metrics for subsequences at the start and the end of the polypeptide. Nevertheless, in order to avoid having a too high-dimensional feature space (which can lead to overfitting for instance), the bipeptide composition was not considered for these subsequences.

In order to select the length of these start and end subsequences, we used cross-validation with the training data. That is, we split the training data into a training and a validation set and tried a range of possible start and end subsequence lengths, training on the new training set and measuring performance on the validation set.

After this cross-validation tuning, we found that it is best to consider thirty aminoacids at the start of the sequence and 20 at the end of the sequence, which is what we proceeded to use in our subsequent experiments.

3.2 Classifiers and Hyperparameter Tuning

After representing the data in our selected feature space, we performed hyper parameter tuning on each of our four classifiers, namely the SVM, the Random Forest, the Gradient boosting and the Logistic regression. We used 5-fold cross validation on each individually in order to do so. The main reasons for this approach (as opposed to tuning all four models simultaneously through the ensembled learner) are: (1) It allows for the different models of the final voting classifier to be independently tuned, hence reducing the risk of overfitting for the ensemble, and (2) It reduces exponentially the time necessary for the process, which from a practicality standpoint allows us to test more combinations of hyperparameters.

The combination of optimal hyperparameters we found from this tuning process is shown in Table 3. The parameters are depicted in a python dictionary format, which is compatible with the implementations of the corresponding models in Scikit-learn.

Table 3. Optimal hyperparameters found through cross-validation, shown in a python dictionary format compatible with Scikit-learn's corresponding model implementations.

Learner	Optimal Hyperparameter Dictionary
SVM	{'kernel': 'rbf', 'C': 0.85}
Gradient Boosting	{'n_estimators': 150, 'subsample': 0.5, 'learning_rate': 0.05, 'max_depth': 4}
Random Forest	{'n_estimators': 800}
Logistic Regression	{'penalty': 'l2', 'C': 1.0, 'max_iter': 300, 'solver': 'newton-cg', 'class_weight': None}

Following the tuning, we make our final model by ensembling these four classifiers in a soft voting scheme, where soft voting means that the final prediction corresponds to the maximum of the sum of probabilities for each class found from each classifier. The ensembled classifier is trained from scratch, using all the hyperparameters previously found.

3.3 Performance and Error Analysis

In order to evaluate the performance of our model, we are using a combination of tests. The main metrics we are using are the accuracy, the F1 score, the precision and the recall. When appropriate, we perform macro-averaging of these metrics over the different classes.

First, we perform a 20-fold cross validation using the training set and calculate the cross-validation mean and standard deviation for the listed metrics. This gives one possible estimate for the generalisation values on these metrics as well as the uncertainty on that estimate. We also report the class-specific F1-score.

We then use the left out test set in order to build a second set of performance metrics. Both of these reported performance metric sets represent independent estimates of their true generalisation values for our model. For a more granular understanding of the performance of the model, we also plot the confusion matrix on the test set, and a set of ROC curves, one for each class and one macro-averaged curve.

Finally, in an attempt to identify common properties or patterns in the features of misclassified examples, we (1) find the most significant features using our decision tree classifiers and (2) for the most important features, we plot the histograms of the fraction of examples that are misclassified as a function of the values of these features.

3.4 Classifying the unknown label examples

As a final task, the examples in the unknown-label (or blind) data are classified using our model. We report the results of the classification, as well as the confidence, which corresponds to the probability for the protein being of the stated class.

4 Results and Discussion

4.1 Performance evaluation Tables

As we discussed in the previous section, as a first step in analysing the performance of our model we calculated a set performance metrics both on the training set (using 10-fold cross validation) and on the left-out test set. Tables 4 and 5 depict these, respectively. On Table 4, we can also see the uncertainty on these metrics, which is obtained by considering two standard deviations on the values obtained from the cross-validation loops.

Table 4. Performance Metrics using 10-fold cross validation on the training set

Evaluation Metric	Value	Uncertainty (2*standard deviation)
Accuracy	0.664	±0.031
Precision	0.702	±0.038
Recall	0.678	±0.037
F1 Score	0.687	±0.035
Cyto F1 Score	0.558	±0.057
Mito F1 Score	0.709	±0.086
Nuc F1 Score	0.669	±0.035
Secr F1 Score	0.812	±0.040

Table 5. Performance metrics on the left out test set

Evaluation Metric	Value
Accuracy	0.680
Precision	0.680
Recall	0.680
F1 Score	0.678
Cyto F1 Score	0.561
Mito F1 Score	0.697
Nuc F1 Score	0.697
Secr F1 Score	0.835

Both tables are quite consistent with each other, and indicate an accuracy of about 67% for our model. We also get that the precision, recall and F1 score are all close to 68 %, both on the cross validation and test set cases. Looking more closely at the F1 scores, it also becomes apparent that the model performs best for the Secreted proteins and worst for the Cytosolic proteins. Finally, looking at Table 4, it is interesting to note that that our classifier seems to have a fairly high variance, of about 4% across different metrics.

4.2 ROC curves and Confusion Matrix

Complementing these results, in Fig. 1 we have plotted the confusion matrix of our model on the test set. Fig. 2 we also show the ROC curves for each of the classes, as well as the corresponding macro-averaged curve.

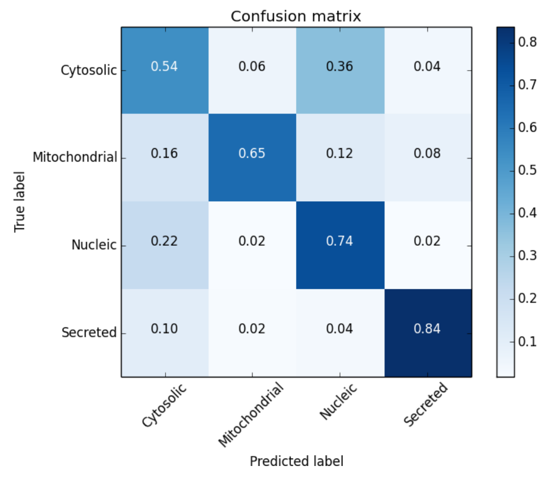

Fig. 1. Confusion Matrix on the test set. The numbers are row-normalised and represent the proportion of examples classified as each class.

The confusion matrix is quite consistent with our conclusions from the performance metrics. Perhaps the uneven performance of the model is even more stark however: We can see that 84% of the secreted proteins are correctly classified, while for the Cytosolic proteins this number is only 54%. Finally, the ROC curves also reinforce the point : they hint to a respectable performance when discriminating Cytosolic proteins (with an area under the ROC curve of 80%), and to great results when discriminating Secreted proteins (97% area under the ROC curve).

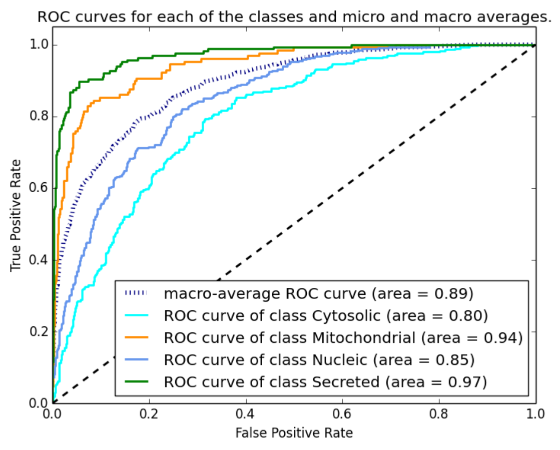

Fig. 2. ROC curves on the test set.

4.3 Error Analysis

In this last part in the analysis of our model's performance, we attempt to draw some more insights into what causes it to make errors in its predictions. To that end, as we mentioned earlier, our first step is to use our decision tree classifiers in order to get a ranking of our best performing features. The top 10 features according to Gradient Boosting are shown in Fig. 3. When considering the Random Forest instead, the results were very similar.

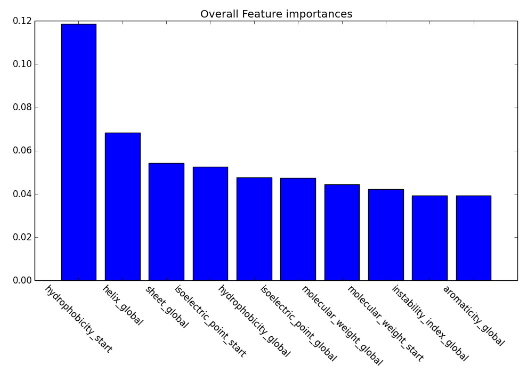

Fig. 3. Histogram of the 10 features discriminating the most between the data, according to the Gradient Boosting classifier.

From Fig. 3, if become apparent that the hydrophobicity at the start of the aminoacid sequence is the most important feature, followed the proportion of aminoacids tending to form different types of secondary structure and the isoelectric point in the start of the sequence.

To see if there are any particular patterns on these four best features that cause the model to fail, we plot the histograms of the feature values of the misclassified test set examples. These histograms are normalised for the frequency of these values on the overall dataset. As such, for each feature value, the bar height corresponds to the proportion of examples having that value that are misclassified. Figs. 4, 5, 6 and 7 show these histograms for the hydrophobicity at the start of the peptide, the overall proportion of α-helix aminoacids, the overall proportion of β-sheet aminoacids and the isoelectric point at the start of the peptide, respectively.

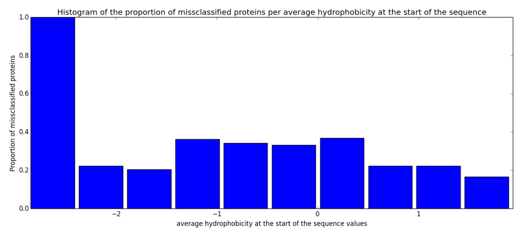

Fig. 4. Histogram of the proportion of misclassified examples for each start-of-the-sequence hydrophobicity value.

No clear trend or strong correlation becomes apparent from these histograms. This is not necessarily counter-intuitive, one would expect examples that are hard to classify to have generic and spread out values for their discriminating features. Nevertheless, interesting notes are (1) it seems that very hydrophilic proteins seem to be disproportionately misclassified, and (2) there appears to be a upwards trend where the more β-sheet aminoacids are present in the peptide the harder it is to classify. These observations are admittedly not conclusive evidence, but arguably can provide interesting pointers for further investigation.

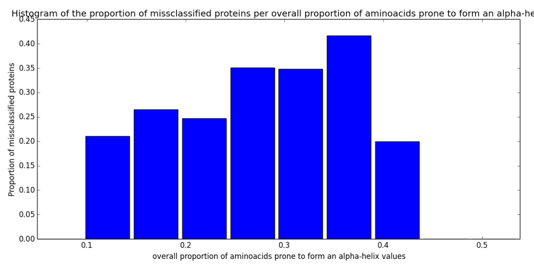

Fig. 5. Histogram of the proportion of the misclassified examples for each value of the proportion of α-helix aminoacids in the peptide.

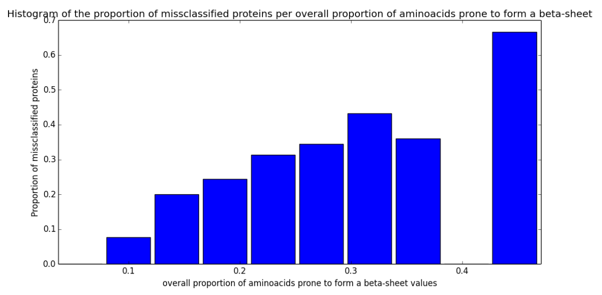

Fig. 6. Histogram of the misclassified examples for each values of the proportion of β-sheet aminoacids in the peptide

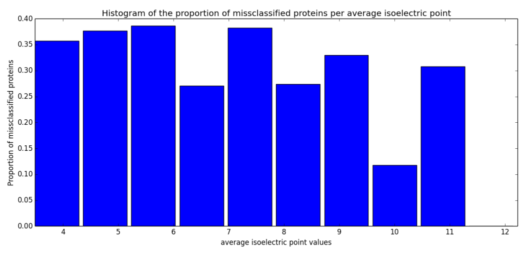

Fig. 7. Histogram of the proportion of misclassified examples for each start-of-the-sequence isoelectric point values.

4.4 Blind Test

Finally, the classification of the proteins on the blind data set is shown in Table 6. This table depicts the identification code of the protein, the class which was assigned to it and the confidence on the classification, which is the probability that the protein is of the specified class.

Table 6. Blind Protein Classification

Protein	Classification	Confidence
SEQ677	Cyto	38.2 %
SEQ231	Mito	47.0 %
SEQ871	Secr	55.9 %
SEQ388	Cyto	44.2 %
SEQ122	Cyto	45.4 %
SEQ758	Nuc	52.5 %
SEQ333	Cyto	43.1 %
SEQ937	Cyto	55.7 %
SEQ351	Cyto	59.2 %
SEQ202	Mito	35.6 %
SEQ608	Cyto	34.8 %
SEQ402	Nuc	46.7 %
SEQ433	Secr	40.0 %
SEQ821	Secr	49.5 %
SEQ322	Nuc	51.5 %
SEQ982	Nuc	53.8 %
SEQ951	Nuc	43.1 %
SEQ173	Cyto	48.9 %
SEQ862	Nuc	38.7 %
SEQ224	Cyto	44.9 %

5 Conclusion

In our investigation, we addressed the protein subcellular location classification task by training an ensemble of learners on the provided data. Through a soft-voting scheme we combined one SVM, one Random Forest, one Logistic Regression and one Gradient Boosting models to make our final classifier. In terms of performance, we estimate that the generalisation accuracy of our model is $66\% \pm 3\%$, which we obtained using ten-fold cross-validation on the training set. We then used a left-out test set of 10% of the data, on which the model achieved an accuracy of 68%. We also run the model on the blind set and depicted the results, although ground truths are not available to measure the corresponding performance.

In order to gain further insights, we also engaged in a more precise error analysis. Using a confusion matrix and ROC curves, we were able to clearly see that Secreted proteins where the best classified, with 84% accuracy, while our model struggled with Cytosolic proteins where only 54% of examples were correctly assigned. Thanks to the decision tree classifiers, we further ranked the features in terms of effectiveness, and found that the hydrophobity at the start of the sequence and the secondary structure indicators were the most powerful. We ended by creating histograms for the proportions of misclassified examples as a function of the values of the best features, but were unable to find any strong correlation between those.

We believe that our classifier gave a honourable performance overall, even though it does not compete with state-of-the-art results. Natural extensions of our work would be to carry this task using Neural Networks and Deep Learning, which allow for the feature representation to be extracted from the data directly. Continued advances in this field are paramount, as the protein localisation task is an important one. It allows deep insights into protein function and by extension to gene function, knowledge that can have profound ramifications, with prevention and treatment of genetic diseases being only one notable example.

Acknowledgements

Professor David Jones for his guidance.
My family and friends for their love and support.

Funding

No external funding has been received for the completion of this study.

References

[1]Charge on proteins. https://people.rit.edu/pac8612/webionex/website/html/ione5svn.html. Accessed : March 2018.

[2]Compgi10 coursework. http://www0.cs.ucl.ac.uk/staff/D.Jones/coursework/. Accessed : March 2018.

[3]Python, python software foundation. https://www.python.org. Accessed : March 2018.

[4]scikit-learn. http://scikit-learn.org/stable/. Accessed : March 2018.

[5]National center for biotechnology information, genome information by organism. https://www.ncbi.nlm.nih.gov/genome/browse/#!/overview/, 2018. Accessed: March 2018.

[6]G. K. Acquaah-Mensah, S. M. Leach, and C. Guda3. Predicting the subcellular localization of human proteins using machine learning and exploratory data analysis. *Genomics, Proteomics & Bioinformatics*, 2(4):120 – 133, 2006.

[7]M. Betts and R. Russell. Amino acid properties and consequences of subsitutions. DIn Bioinformatics for Geneticists, M.R. Barnes, I.C. Gray eds, Wiley, 2003. Accessed : March 2018.

[8]biochemistryquestions. Secondary structure of proteins. The Biochemistry Questions Site, https://biochemistryquestions.wordpress.com/2008/10/02/secondary-structure-of-proteins/, 2008. Accessed : March 2018.

[9]BiologyWise.com. The structure and functions of a cell nucleus explained. https://biologywise.com/cell-nucleus-structure-functions, 2018. Accessed : March 2018.

[10]BioPythobn. Class proteinanalysis. http://biopython.org/DIST/docs/api/Bio.SeqUtils.ProtParam.ProteinAnalysis-class.html#aromaticity. Accessed : March 2018.

[11]ChemCollective. Stoichiometry tutorials: Calculating molecular weight / molar mass. http://chemcollective.org/activities/tutorials/stoich/calculating_molecular_weight. Accessed : March 2018.

[12]K.-C. Chou and H.-B. Shen. A new method for predicting the subcellular localization of eukaryotic proteins with both single and multiple sites: Euk-mploc 2.0. *Martin DP, ed. PLoS ONE. 2010;5(4):e9931*, 2010.

[13]A. Day. Definition of stability. http://www.cryst.bbk.ac.uk/PPS2/projects/day/TDayDiss/StabilityDefined.html, 1996. Accessed : March 2018.

[14]P. DÃ¶nnes and A. HÃ¶glund. Predicting protein subcellular localization: Past, present, and future. *Genomics, Proteomics & Bioinformatics*, 2(4):209 – 215, 2004.

[15]A. GÃ©ron. *Hands-On Machine Learning with Scikit-Learn and TensorFlow*. O'Reilly Media, 3 2017.

[16]I. Hunt. Secreted proteins. Department of Chemistry, University of Calgary, http://www.chem.ucalgary.ca/courses/351/Carey5th/Ch27/ch27-1-4.html. Accessed : March 2018.

[17]J. Kyte, Russell, and F. Doolittle. A simple method for displaying the hydropathic character of a protein. *Journal of Molecular Biology*, 1982.

[18]M. MBU. What are mitochondria? Mitochondrial Biology Unit, Cambridge University, http://www.mrc-mbu.cam.ac.uk/what-are-mitochondria, 2018. Accessed : March 2018.

[19]H. Nakashima and K. Nishikawa. Discrimination of intracellular and extracellular proteins using amino acid composition and residue-pair frequencies. *Journal of Molecular Biology*, 238(1):54–61, 1994.

[20]proteinatlas.org. Secreted proteins. The Human Protein Atlas, https://www.proteinatlas.org/humancell/secreted+proteins. Accessed : March 2018.

[21]Study.com. Cytosol: Definition, function & structure. https://study.com/academy/lesson/cytosol-definition-function-structure.html, 2018. Accessed : March 2018.

[22]S. Wan and Q. Zou. Hpslpred: An ensemble multi-label classifier for human protein subcellular location prediction with imbalanced source. *CoRR*, abs/1704.05204, 2017.

[23]Wikipedia. Amino acid. https://en.wikipedia.org/wiki/Amino_acid#cite_note-Hausman-136. Accessed : March 2018.